(12) UK Patent (19) GB (11) 2 063 532 B

(54) Title of invention

Data storage system for a computer

(51) INT CL³: G06F 13/00

(21) Application No
8029863

(22) Date of filing
16 Sep 1980

(30) Priority data

(31) 85909

(32) 18 Oct 1979

(33) United States of America (US)

(43) Application published
3 Jun 1981

(45) Patent published
16 May 1984

(73) Proprietor
Storage Technology Corporation,
P.O. Box 98
Louisville
Colorado 80027
United States of America

(72) Inventor
Barry B. White

(74) Agent and/or Address for Service
Reddie & Grose,
16 Theobalds Road
London WC1X 8PL

(52) Domestic classification
G4A MX

(56) Documents cited
GB 1547381
GB 1496780
GB 1496779
GB 1359662
GB 1353770
GB 1167762

(58) Field of search
G4A

LONDON THE PATENT OFFICE

GB 2 063 532 B

2063532

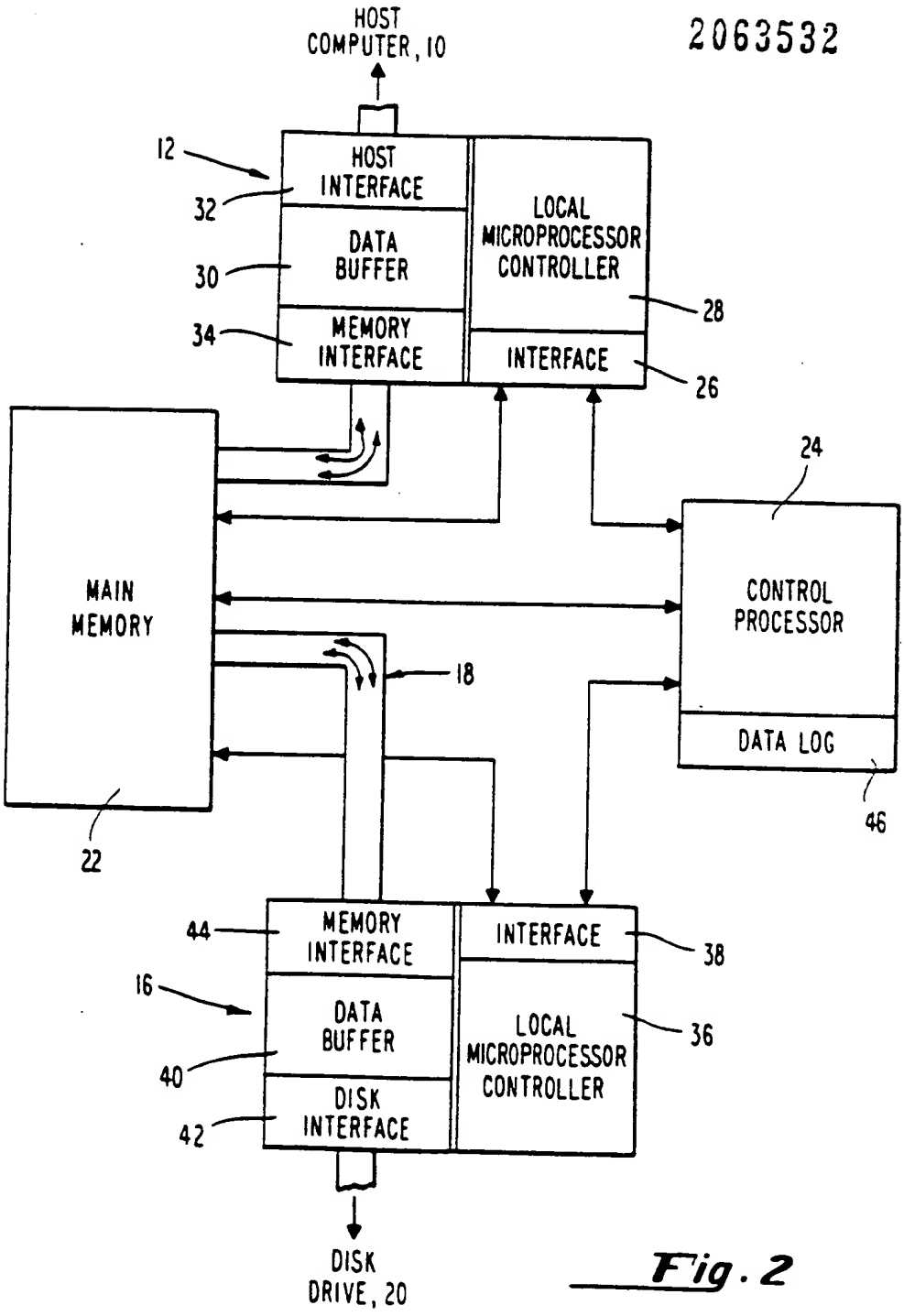

Fig. 2

STK V. EMC
STK 02868

2063532

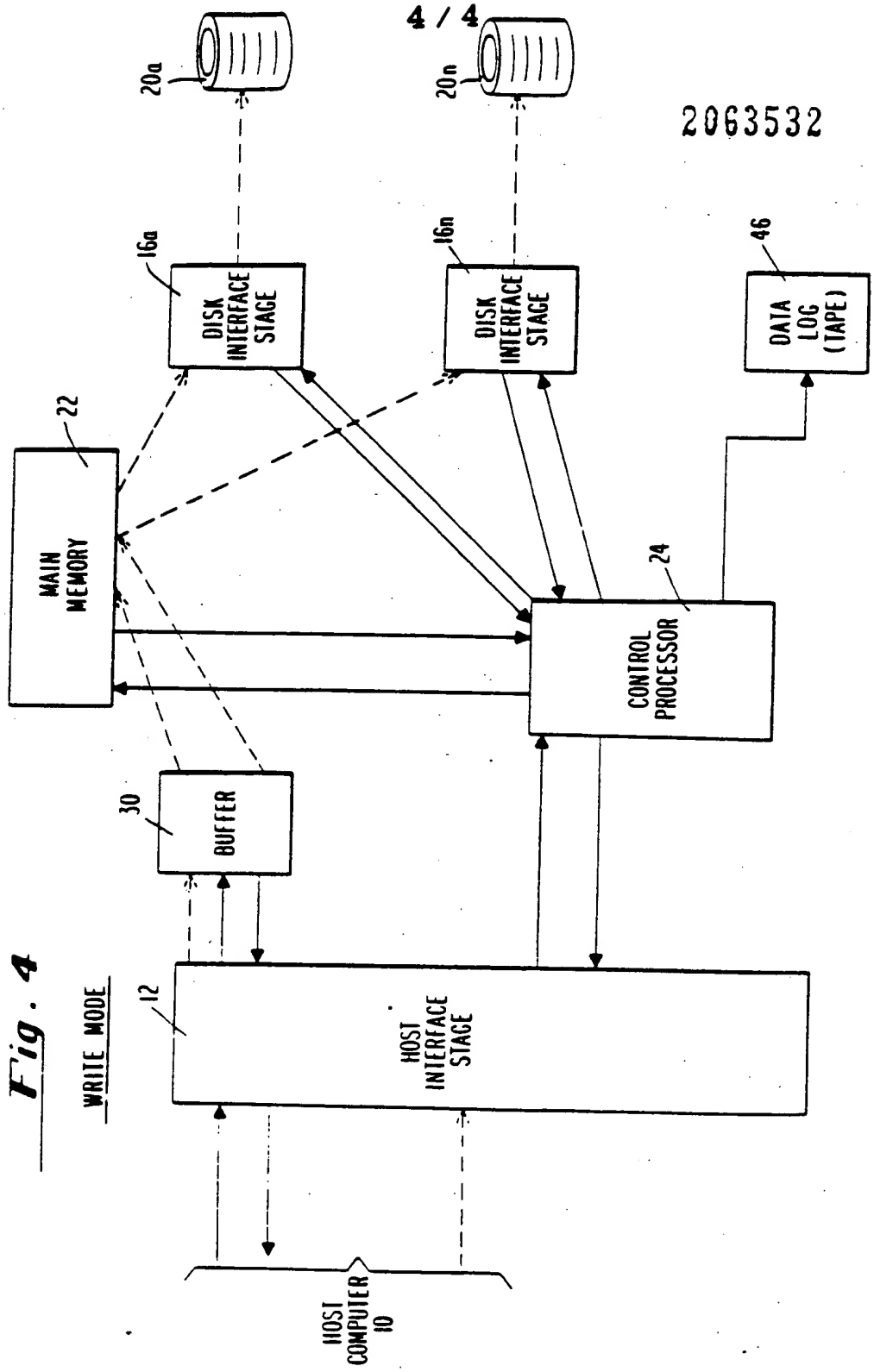

Fig. 4

WRITE MODE

thousands of reels of tape. In general, some 90% of all tapes contain but one data set (e.g. a customer list or a payroll file) since as a rule the physical characteristics of tape make it impractical to read or write from or to two or more data sets at once, which is very often desirable. Clearly then, tape storage is very inefficient, since only coincidentally will the length of a given data approximate the available memory area. In fact, studies show that normally only about one percent of tape is actually used.

Obviously, no user can provide a tape mount or drive for each reel of tape; rather, individual reels are constantly brought from storage, mounted for use, then demounted and replaced in storage. This is time-taking and inefficient, since it involves the intervention of a human operator.

Efforts have been made to overcome this shortcoming of tape media, for instance by providing a large number of small cartridges of tape and an automatic mechanism to rapidly remove the cartridges from a storage rack and place them on a tape mount. Automatic mounting devices have also been made for use with conventional tapes. To date, however, no completely successful method exists for making large quantities of tape immediately available to tape drives.

At first glance, disk storage system would appear to overcome the shortcomings of tape. Practically any portion of any disk can be assessed very rapidly, so that a disk system does not have to scan through all of the data recorded thereon in serial fashion. Accordingly, by responding to address codes, a disk drive can access a given block of data almost instantly. Thus, multiple data sets can be stored on a single disk and be concurrently processed. However, the cost of disk drives is comparatively great, and moreover available disk drives do not make highly efficient use of the potential storage area.

Studies have demonstrated that at any given time in a disk system, only approximately 75% of potential storage space is actually available for data storage purposes. The "overhead",

stages respond to the host computer in the same manner as would a tape system, while the disk interface stages read and write data upon disk memories in random fashion in accordance with currently available memory space.

The invention will be described in more detail, by way of example, with reference to the accompanying drawings in which:

Figure 1 illustrates the overall organization of a virtual storage system embodying the invention;

Figure 2 is a schematic diagram illustrating the flow of control and information signals within the system of Figure 1; and

Figures 3 and 4 direct the flow of command signals and information in "read" and "write" situations, respectively.

Figure 1 shows in simplified form the basic architecture of the virtual storage system embodying the invention. The system provides means for controlling one or more extremely fast and accessible disk memory units to accept data and respond in precisely the same fashion as a tape drive, while providing much better use of available storage, thus in effect having vastly expanded storage capability and a far more rapid response time than either tape or disks. The host computer 10 may be of any appropriate type, although it is anticipated that economical operation of the present invention will dictate it use with a large mainframe host computer such as IBM System/360 and System/370 computers, and IBM Model 3031, 3032 and 3033 processor complexes (IBM is a Registered Trade Mark). The virtual storage system can also be used with a plurality of host computers. The storage system comprises a bank of host interface stages 12_a to 12_n, the number of interface stages being dependent upon the amount of data to be transferred. Since the host interface stages are identical, the actual number which is to be used is for present purposes immaterial. The host interface stages are coupled to the host computer 10 by means of conventional channels 14 and receive and transmit information to the computer 10 in the ordinary, serialized format which is associated with tape drives.

One or more disk interface stages 16_a to 16_n are provided, and are coupled to all of the host interface stages by means of a common information bus 18. In this manner, information can be

Such signals are obtained by coupling the host interface stage to the byte multiplex, or block multiplex, or selector channels of the host computer 10. It will be appreciated that these signals include operator commands such as "mount tape" and "demount tape" as well as machine commands such as "read", "write", and "forward space file". The host interface stage responds to both kinds of signals as if it were an operator and a tape drive, acknowledging the signals, responding that "mounting" of an imaginary (virtual) tape reel has been accomplished, and so on.

The data buffer 30 of each host interface stage accepts data from the host computer 10 in serial form, usually nine bits in parallel, in the precise manner that it would be reapplied to a tape system for writing upon a tape. The buffer 30 in conjunction with the host interface 32 during a write operation de-serializes the information, that is stores it in parallel fields holding the individual bits until eight entire bytes are available. Typically 72 bits are transmitted at a time thus reducing transmission time. In this manner, up to 90% compression of the time required to transmit the data on the line 18 can be achieved. When data is to be exchanged between the buffer 30 and the bus 18, the data flow through the memory interface 34, again in eight parallel bytes so that an extremely rapid exchange of data may take place. The size of the buffer can vary greatly, depending on system requirements but in a preferred embodiment is large enough, 64,000 bytes, to hold an entire record, which expedites time sharing of bus 18.

Ultimately, data received from the host computer 10 is written on to magnetic disks mounted on one or more disk drive units 20_a to 20_n. Each disk interface stage 16 couples the data bus 18 to a disk drive 20 (not shown), and is composed of a local microprocessor controller 36 which operates the various elements of the interface stage in accordance with instructions from the control processor 24. As was the case with the host interface stages 12, with disk interface stages 16 instructions from the control processor are transmitted through an interface 38 to the local controller 36. The latter then responds by causing a data buffer 40, disk interface 42, and memory interface 44 to transfer data to and from the disk drive.

In an alternative embodiment, both the host buffers 30 and disk buffers 40 may be dispensed with, their functions then being performed by main memory 22.

In this fashion, segments of data which had been dispersed throughout a number of disk drives are compiled, queued, and then re-assembled automatically into serial form. Accordingly, the information flowing to the host computer 10 from the various disk drives 20 appears in serial form, precisely as if it were being read from a tape. In the foregoing manner, the high speed main memory provides a cache from which data can be selected and queued for subsequent reassembly into serial form before being directed to the host computer.

As will be recognized by those skilled in the art, the various elements of the present virtual storage system may be assembled from commercially-available elements, and coupled together in any convenient fashion. For instance, while all of the various elements of disk interface stage 16 are illustrated as being disposed in one location, remote from control processor 24, in fact the elements may be disposed at different locations, and coupled by means of appropriate cables, buses, or the like. The local microprocessor controllers 24 used for operating the components of each interface stage need only be of rather limited capability and may be any of the various high speed microprocessors available on the market. An example of such microprocessors is the LSI-11 marketed by DEC Incorporated of Boston, Mass, alternatively, a suitable unit can be assembled from the AMD Co's 2900 series of parts. In like manner, the host and disk interfaces 12 and 16 which accomplish the serializing and deserializing of information may be standard units, such as the model 370 block multiplexer available from the IBM Corporation of Armonk, New York. In like manner, an IBM block multiplexer may be used for direct memory access, the actual connection of the various units being well understood by those skilled in the art.

Similarly, the buffers 30 and 40 used for temporary storage of data in the host and disk interface stages 12 and 16 may be of any appropriate type, although in the embodiment at present preferred, a memory of at least 64K bytes is preferred. One commercially available buffer of this type is manufactured by Fairchild Semiconductor, and comprises a N-MOS random access memory having

the specific identity of the block of information to be read. This command is transmitted directly by the host interface stage 12 to the control processor 24 which identifies the various areas in which the information is stored. This may be accomplished through the use of a data log 46 which is coupled to the control processor. In the preferred embodiment of the data log 46 comprises a random access memory (RAM) (which may be "backed-up" by a tape drive unit and associated tape in case of loss of memory in the RAM) on which the location of the various sub-blocks of data which together constitute the data requested by the host computer 10 are recorded.

When the locations of the various sub-blocks of data have been identified to the control processor 24, signals are supplied to the controller portions of the various disk interface stages 36, whereupon data is accessed from the various disk storage units 20. The data is then read out into the buffer 40 of the interface stage 16 and prepared for transmission to the main memory cache 22 for temporary residence there until needed by the various host interface stages 12.

When the buffer 40 is filled with data, a signal is output from the disk interface stage 16 to the control processor 24 indicating this fact, at which time the control processor 24 transmits the buffered information directly to a previously-allocated space in the main, high speed memory cache 22.

With prior art tape systems, the initial "Mount" operation ordinarily occupies approximately 30 sec. - 5 min. The total time required to execute a "read" command varies between 1-10 milliseconds depending on the characteristics of the individual device and of course on the length of the record being read. In a preferred embodiment of the illustrated virtual storage system, a "Mount" operation typically occupies less than one second. The actual "read" operation typically occupies 1-10 milliseconds as before, depending on record length and the characteristics of the cache 22. At this point, therefore, the information to have been read will have been very rapidly placed in the cache 22 where it is stored until a signal from memory interface 12 indicates to the cache that the host computer is ready to receive the data. At this stage, then, the data has been gathered from the various points in the various disks 20 where it

22. At some subsequent time, when adequate buffer capacity is available in one or more disk interface stages 16, the control processor instructs the cache memory 22 to supply the stored data to the appropriate disk interface stage buffers 16 for disposition upon the disks of the associated disk drives 20.

As each block of data is applied to a host interface 32 by the host 10, the memory interface module 34 determines where in the associated buffer 30 the newly-received information will be stored; compresses the data in serial fashion so as to produce, for instance, eight channels of data; and adds an identification bit or bits to the information block indicating the size of the block. In this manner, data is continually fed to the buffers 30 until a given block of data is filled or until the host computer generates a signal indicating the end of a given record. It is an important subsidiary feature of the present invention that such a signal is interpreted by the local controller 28 as an end of message signal, subsequent to which no more buffer memory space need be allocated.

At this point, the control processor 24 directs the data now stored in data buffer 30 to the high-speed cache 22. During the time the host computer is storing data in data buffer 30 as described above, the control processor 24 can be "looking for" sufficient space upon disk drives 20, and can allocate that space to the data then being stored in data buffer 30. Thus, when the end of record signal is received in the control processor, data it can be serialized and fed into the high speed cache, as a temporary storage space, and is then passed to disk drives 20 for storage.

Put differently, when the host computer produces an initial "Mount Tape" command (i.e. directs the operator to provide a blank tape for data storage purposes) this command is passed through the host interface stage 12 to control processor 24. The control processor 24 responds to the "Mount Tape" instruction by seeking space upon one or more disks 20, and by reserving adequate space in one or more of the disk interface stage buffers 30. This done, the control processor 24 instructs the host interface stage 12 to accept data from the host 10 and to deserialize it and store

efficiency and eliminate undue multiplication of disk or tape storage systems. The host interface units of the invention may be coupled to more than one host computer. Moreover, it will be appreciated that the provision of a high speed memory or cache is essential to the "anticipatory buffering" which is a very important feature. By using the high speed cache as an intermediate buffer between the host computer interface and the various disk drive interface stages, buffering of multiplex data can be accomplished; that is, data stored in a number of places upon a number of disk drives (referred to as "in random form") can be assembled, sequenced, and stored temporarily in the high speed cache until it is called for by the host computer. In this way, there need be no delays in reading the information from memory into the host computer. In a similar fashion, data output from the host computer can be arranged, divided, and stored in the high speed cache until such time as various disk storage areas are available to store the information.

A particularly significant possibility which is provided by the storage system is that of data compression. Heretofore a common scheme of data compression, which may involve the replacement of a long string of digital "1's" or "0's" with symbols indicating the length of the string, had not been useful with disk drive units, since this scheme compresses addressing information as well as data. However, since according to the present application disk drives appear as tape, this difficulty of the prior art is obviated, and data compression is made compatible with disk storage. Preferably, data compression is implemented in the host interface stage; that is during a write operation long continuous strings of "1's" or "0's" may be detected and replaced with shorter symbols; during a read operation, these are detected and replaced by the thus-defined data.

Finally, it will be appreciated that there are a number of other modifications and refinements which can be made to the virtual storage system of the invention and that the examples discussed above are exemplary only.

serial form, and for causing the or each second interface to storage and to access information in random form, as hereinbefore defined.

3. A data storage system as claimed in either of claims 1 and 2, wherein the system is responsive to both operator and machine commands.

4. A data storage system as claimed in either of claims 1 and 2, further comprising means for effecting data compression and data decompression.

5. A data storage system as claimed in claim 1, wherein each first interface comprises a data buffer.

6. A data storage system as claimed in claim 5, wherein each second interface comprises a data buffer.

7. A data storage system according to either of claims 1 and 2, wherein the data received from a computer by a first interface is de-serialized prior to transmission to the main memory, and is re-serialized prior to being written on to the disk memory devices.

8. A data storage system according to claim 7, wherein the serially organized data is additionally stored on magnetic tape memory means.

9. A data storage system according to either of claims 1 and 2, wherein the addresses of the storage locations at which the data is stored on the disk devices are stored in a random access memory.

10. A data storage system according to claim 8, wherein the addresses are additionally stored in back-up memory means.